Requirements Prioritization Introduction

Nancy R. Mead, Software Engineering Institute [vita[1]]

2006-09-26; Updated 2008-09-18 L3 / L[2]

Once you have identified a set of security requirements, you will usually want to prioritize them. Due to time and budget constraints, it can be difficult to implement all requirements that have been elicited for a system. Also, security requirements are often implemented in stages, and prioritization can help to determine which ones should be implemented first. Many organizations pick the lowest cost requirements to implement first, without regard to importance. Others pick the requirements that are easiest to implement, for example by purchasing a COTS solution. These ad hoc approaches are not likely to achieve the security goals of the organization or the project. To prioritize security requirements, we recommend a systematic prioritization approach. This article discusses a tradeoff analysis that you can do to select a suitable requirements prioritization method and briefly describes a number of methods. A companion case study [Chung 06][3] can be found in Requirements Prioritization Case Study Using AHP[4]. While results may vary for your organization, the discussion of the various techniques should be of interest. Much work needs to be done before security requirements prioritization is a mature area, but it is one that we must start to address.

Identify Candidate Prioritization Methods

A number of prioritization methods have been found to be useful in traditional requirements engineering and could potentially be used for security requirements. We briefly mention here the Binary Search Tree[10], Numeral Assignment Technique[11], Planning Game[12], the 100-Point Method[13], Theory-W[14], Requirements Triage[15], Wiegers' Method[16], Requirements Prioritization Framework[17], and AHP[18].

Binary Search Tree (BST)

Binary Search Tree is an algorithm that is typically used in a search for information and can easily be scaled to be used in prioritizing many requirements [Ahl 05[19]]. The basic approach for requirements is as follows, quoting from [Ahl 05]:

1. Put all requirements in one pile.

[1] http://buildsecurityin.us-cert.gov/bsi/about_us/authors/230-BSI.html (Mead, Nancy)
[3] #dsy545-BSI_Chung
[4] http://buildsecurityin.us-cert.gov/bsi/articles/best-practices/requirements/534-BSI.html (Requirements Prioritization Case Study Using AHP)
[5] #dsy545-BSI_icpm
[6] #dsy545-BSI_ptc
[7] #dsy545-BSI_recs
[8] #dsy545-BSI_ack
[9] #dsy545-BSI_refs
[10] #dsy545-BSI_BST
[11] #dsy545-BSI_NAT
[12] #dsy545-BSI_PG
[13] #dsy545-BSI_OPM
[14] #dsy545-BSI_TW
[15] #dsy545-BSI_RT
[16] #dsy545-BSI_WM
[17] #dsy545-BSI_RPF
[18] #dsy545-BSI_AHP
[19] #dsy545-BSI_Ahl

2. Take one requirement and put it as root node.
3. Take another requirement and compare it to the root node.
4. If the requirement is less important than the root node, compare it to the left child node. If the requirement is more important than the root node, compare it to the right child node. If the node does not have any appropriate child nodes, insert the new requirement as the new child node to the right or left, depending on whether the requirement is more or less important.
5. Repeat steps 3-4 until all requirements have been compared and inserted into the BST.
6. For presentation purposes, traverse through the entire BST in order and put the requirements in a list, with the least important requirement at the end of the list and the most important requirement at the start of the list.

Numeral Assignment Technique

The Numeral Assignment Technique provides a scale for each requirement. Brackett proposed dividing the requirements into three groups: mandatory, desirable, and unessential [Brackett 90][20]. Participants assign each requirement a number on a scale of 1 to 5 to indicate its importance [Karlsson 95][21]. The numbers carry the following meaning:

1. does not matter (the customer does not need it)
2. not important (the customer would accept its absence)
3. rather important (the customer would appreciate it)
4. very important (the customer does not want to be without it)
5. mandatory (the customer cannot do without it)

The final ranking is the average of all participants' rankings for each requirement.

Planning Game

The planning game is a feature of extreme programming [Beck 04][22] and is used with customers to prioritize features based on stories. This is a variation of the Numeral Assignment Technique, where the customer distributes the requirements into three groups, “those without which the system will not function,” “those that are less essential but provide significant business value,” and “those that would be nice to have.”

100-Point Method

The 100-Point Method [Leffingwell 03][23] is basically a voting scheme of the type that is used in brainstorming exercises. Each stakeholder is given 100 points that he or she can use for voting in favor of the most important requirements. The 100 points can be distributed in any way that the stakeholder desires. For example, if there are four requirements that the stakeholder views as equal priority, he or she can put 25 points on each. If there is one requirement that the stakeholder views as having overarching importance, he or she can put 100 points on that requirement. However, this type of scheme only works for an initial vote. If a second vote is taken, people are likely to redistribute their votes to get their favorites moved up in the priority scheme.

Theory-W

Theory-W was initially developed at the University of Southern California in 1989 [Boehm 89[24], Park 99][25]. It is also known as "Win-Win." An important point is that it supports negotiation to solve disagreements about requirements, so that each stakeholder has a "win." It has two principles:

[20] #dsy545-BSI_Brackett
[21] #dsy545-BSI_Karlsson95
[22] #dsy545-BSI_Beck
[23] #dsy545-BSI_Leff
[24] #dsy545-BSI_Boehm

1. Plan the flight and fly the plan.
2. Identify and manage your risks.

The first principle seeks to build well-structured plans that meet predefined standards for easy development, classification, and query. “Fly the plan” ensures that the progress follows the original plan. The second principle, “Identify and manage your risks,” involves risk assessment and risk handling. It is used to guard the stakeholders’ “win-win” conditions from infringement. In win-win negotiations, each user should rank the requirements privately before negotiations start. In the individual ranking process, the user considers whether there are requirements that he or she is willing to give up on, so that individual winning and losing conditions are fully understood. Theory-W has four steps:

1. Separate the people from the problem.
2. Focus on interests, not positions.
3. Invest options for mutual gain.
4. Insist on using objective criteria.

Requirements Triage

Requirements Triage [Davis 03][26] is a multistep process that includes establishing relative priorities for requirements, estimating resources necessary to satisfy each requirement, and selecting a subset of requirements to optimize probability of the product’s success in the intended market. This is clearly aimed at developers of software products in the commercial marketplace. Davis’s more recent book [Davis 05][27] expands on the synergy between software development and marketing; we recommend that you read it if you are considering this approach. It is a unique approach that is worth reviewing, although it clearly goes beyond traditional requirements prioritization, considering business factors as well.

Wiegers' Method

This method relates directly to the value of each requirement to a customer [Wiegers 03][28]. The priority is calculated by dividing the value of a requirement by the sum of the costs and technical risks associated with its implementation [Wiegers 03][29]. The value of a requirement is viewed as depending on both the value provided by the client to the customer and the penalty that occurs if the requirement is missing. This means that developers should evaluate the cost of the requirement and its implementation risks, as well as the penalty incurred if the requirement is missing. Attributes are evaluated on a scale of 1 to 9.

Requirements Prioritization Framework

The requirements prioritization framework and its associated tool [Moisiadis 00[30], Moisiadis 01][31] includes both elicitation and prioritization activities. This framework is intended to address the following:

- elicitation of stakeholders' business goals for the project
- rating the stakeholders using stakeholder profile models
- allowing the stakeholders to rate the importance of the requirements and the business goals using a fuzzy graphic rating scale
- rating the requirements based on objective measure
- finding the dependencies between the requirements and clustering requirements so as to prioritize them more effectively

[25] #dsy545-BSI_Park
[26] #dsy545-BSI_Davis03
[27] #dsy545-BSI_Davis05
[28] #dsy545-BSI_Wiegers
[29] #dsy545-BSI_Wiegers
[30] #dsy545-BSI_Mois00
[31] #dsy545-BSI_Mois01

- using risk analysis techniques to detect cliques among the stakeholders, deviations among the stakeholders for the subjective ratings, and the association between the stakeholders' inputs and the final ratings

AHP

AHP is a method for decision making in situations where multiple objectives are present [Saaty 80[32], Karlsson 96[33], Karlsson 97][34]. This method uses a "pair-wise" comparison matrix to calculate the relative value and costs of individual security requirements to one another. By using AHP, the requirements engineer can confirm the consistency of the result. AHP can prevent subjective judgment errors and increase the likelihood that the results are reliable. AHP is supported by a standalone tool, as well as by a computational aid within the SQUARE tool.There are five steps in the AHP method:

1. Review candidate requirements for completeness.
2. Apply the pair-wise comparison method to assess the relative value of each of the candidate requirements.
3. Apply the pair-wise comparison method to assess the relative cost of the candidate requirements.
4. Calculate each candidate requirement's relative value and implementation cost, and plot each on a cost-value diagram.
5. Use the cost-value diagram as a map for analyzing the candidate requirement.

Prioritization Technique Comparison

We recommend comparing several candidate prioritization techniques to aid in selecting a suitable technique. Some example evaluation criteria are

- **clear-cut steps:** There is clear definition between stages or steps within the prioritization method.
- **quantitative measurement:** The prioritization method's numerical output clearly displays the client's priorities for all requirements.
- **high maturity:** The method has had considerable exposure and analysis in the requirements engineering community.
- **low labor-intensity:** A reasonable number of hours are needed to properly execute the prioritization method.
- **shallow learning curve:** The requirements engineers and stakeholders can fully comprehend the method within a reasonable length of time.

Note that this simple approach does not consider the importance of each criterion. It is also possible to do a weighted average when comparing techniques. For example, maturity may be of more importance than learning curve. This could be taken into account by weighting the results and ranking the various criteria as "essential" with weight 3, "desirable" with weight 2, and "optional" with weight 1. A comparison matrix used in a case study is shown in Table 1[35]. This is not intended to be an actual recommendation to use a specific technique; you can develop your own comparison criteria and ratings.

For one of our case studies we considered the Numeral Assignment Technique (NAT), Theory-W (TW), and AHP. The results of the **comparison** are summarized in Table 1.

Table 1. Comparison of prioritization techniques for a case study
3= Very Good, 2= Fair, 1= Poor

	Numeral Assignment Technique	**Theory-W**	**AHP**

[32] #dsy545-BSI_Saaty
[33] #dsy545-BSI_Karlsson96
[34] #dsy545-BSI_Karlsson97
[35] #dsy545-BSI_tbl1

clear-cut steps	3	2	3
quantitative measurement	3	1	3
maturity	1	3	3
labor-intensive	2	1	2
learning curve	3	1	2
Total score	12	8	16

We decided to use AHP as a prioritizing method. This was done on the basis of the above comparison, recognizing that the rankings are subjective. Factoring into the rationale behind choosing AHP were the team members' familiarity with the method, its quantitative outputs, and its structure in providing definite steps for implementation. The detailed case study results are described in Requirements Prioritization Case Study Using AHP[36].

Recommendations for Requirements Prioritization

Prioritization of security requirements is an important activity. We recommend that stakeholders select candidate prioritization techniques, develop selection criteria to pick one, and apply it to decide which security requirements to implement when. During the prioritization process, the stakeholders can verify that everyone has the same understanding about the security requirements and further examine any ambiguous requirements. After everyone reaches consensus, the results of the prioritization exercise will be more reliable.

Acknowledgement

This material is primarily extracted and adapted from a more extensive report by Lydia Chung, Frank Hung, Eric Hough, and Don Ojoko-Adams [Chung 06][37].

References

[Ahl 05] Ahl, V. "An Experimental Comparison of Five Prioritization Methods." Master's Thesis, School of Engineering, Blekinge Institute of Technology, Ronneby, Sweden, 2005.

[Beck 04] Beck, K. & Andres, C. *Extreme Programming Explained: Embrace Change*, 2nd ed. Boston, MA: Addison-Wesley, 2004.

[Boehm 89] Boehm, B. & Ross, R. "Theory-W Software Project Management: Principles and Examples." *IEEE Transactions on Software Engineering 15*, 4 (July 1989): 902-916.

[Brackett 90] Brackett, J. W. *Software Requirements*[38] (SEI-CM-19-1.2, ADA235642). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 1990.

[36] http://buildsecurityin.us-cert.gov/bsi/articles/best-practices/requirements/534-BSI.html (Requirements Prioritization Case Study Using AHP)

[37] #dsy545-BSI_Chung

[Chung 06] Chung, L.; Hung, F.; Hough, E.; & Ojoko-Adams, D. *Security Quality Requirements Engineering (SQUARE): Case Study Phase III*[39] (CMU/SEI-2006-SR-003). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 2006.

[Davis 03] Davis, A. "The Art of Requirements Triage." *IEEE Computer 36*, 3 (March 2003): 42-49.

[Davis 05] Davis, A. *Just Enough Requirements Management: Where Software Development Meets Marketing*. New York: Dorset House, 2005 (ISBN 0-932633-64-1).

[Karlsson 95] Karlsson, J. "Towards a Strategy for Software Requirements Selection. Licentiate." Thesis 513, Linköping University, October 1995.

[Karlsson 96] Karlsson, J. "Software Requirements Prioritizing," 110-116. *Proceedings of the Second International Conference on Requirements Engineering (ICRE'96).* Colorado Springs, CO, April 15-18, 1996. Los Alamitos, CA: IEEE Computer Society, 1996.

[Karlsson 97] Karlsson, J. & Ryan, K. "A Cost-Value Approach for Prioritizing Requirements." *IEEE Software 14,* 5 (September/October 1997): 67-74.

[Leffingwell 03] Leffingwell, D. & Widrig, D., *Managing Software Requirements: A Use Case Approach,* 2nd ed. Boston, MA: Addison-Wesley, 2003.

[Lehtola 04] Lehtola, L. & Kauppinen, M. "Empirical Evaluation of Two Requirements Prioritization Methods in Prodcut Development Projects," 161-170. *EuroSPI 2004*, LNCS 3281. Heidelberg, Germany: Springer-Verlag, 2004.

[Moisiadis 00] Moisiadis, F. "Prioritising Scenario Evolution." *International Conference on Requirements Engineering (ICRE 2000)*. June 2000.

[Moisiadis 01] Moisiadis, F. "A Requirements Prioritisation Tool." *6th Australian Workshop on Requirements Engineering (AWRE 2001)*. Sydney, Australia, November 2001.

[Park 99] Park, J.; Port, D.; & Boehm B. "Supporting Distributed Collaborative Prioritization for Win-Win Requirements Capture and Negotiation," 578-584. *Proceedings of the International Third World Multi-conference on Systemics, Cybernetics and Informatics (SCI'99) Vol. 2*. Orlando, FL, July 31-

August 4, 1999. Orlando, FL: International Institute of Informatics and Systemics (IIIS), 1999.

[Saaty 80] Saaty, T. L. *The Analytic Hierarchy Process*. New York, NY: McGraw-Hill, 1980.

[Wiegers 03] Wiegers, K. E. *Software Requirements*, 2nd ed. Redmond, WA: Microsoft Press, 2003.

Carnegie Mellon Copyright

The Build Security In (BSI) portal is sponsored by the U.S. Department of Homeland Security (DHS), National Cyber Security Division. The Software Engineering Institute (SEI) develops and operates BSI. DHS funding supports the publishing of all site content.

NO WARRANTY

THIS MATERIAL OF CARNEGIE MELLON UNIVERSITY AND ITS SOFTWARE ENGINEERING INSTITUTE IS FURNISHED ON AN “AS-IS" BASIS. CARNEGIE MELLON UNIVERSITY MAKES NO WARRANTIES OF ANY KIND, EITHER EXPRESSED OR IMPLIED, AS TO ANY MATTER INCLUDING, BUT NOT LIMITED TO, WARRANTY OF FITNESS FOR PURPOSE OR MERCHANTABILITY, EXCLUSIVITY, OR RESULTS OBTAINED FROM USE OF THE MATERIAL. CARNEGIE MELLON UNIVERSITY DOES NOT MAKE ANY WARRANTY OF ANY KIND WITH RESPECT TO FREEDOM FROM PATENT, TRADEMARK, OR COPYRIGHT INFRINGEMENT.

[1] mailto:permission@sei.cmu.edu